Contents

CONTEXT SR0_Path

SETS

PATH

ROUTE

CONSTANTS

PathConflict

Route2InitPath

PathSub

NullPath

AXIOMS

axm1: $PathConflict \in PATH \leftrightarrow PATH$

axm2: $\forall p \cdot p \in PATH \Rightarrow (PathConflict \cap id = \varnothing)$

axm3: $\forall p \cdot p \in PATH \Rightarrow (PathConflict = PathConflict^{-1})$

axm4: $\forall p \cdot p \in PATH \Rightarrow (finite(PATH))$

axm5: $Route2InitPath \in ROUTE \rightarrow PATH$

axm6: ⟨theorem⟩ $PathSub \in PATH \leftrightarrow PATH$

axm7: $\forall p \cdot p \in PATH \Rightarrow (\forall p1, p2 \cdot p1 \in PathSub[\{p2\}] \land p \notin PathConflict[\{p2\}] \Rightarrow p \notin PathConflict[\{p1\}])$

axm8: $\forall p \cdot p \in PATH \Rightarrow NullPath \in PathSub[\{p\}]$

axm9: $NullPath \in PATH$

END

CONTEXT SR1_Block
EXTENDS SR0_Path
SETS
BLOCK
CONSTANTS
Path2Block
PathReduce
PathIncrease
AXIOMS
axm1: $Path2Block \in PATH \leftrightarrow BLOCK$
axm2: $\forall p \cdot p \in PATH \Rightarrow (\forall q \cdot q \notin PathConflict[\{p\}] \Leftrightarrow (Path2Block[\{p\}] \cap Path2Block[\{q\}] = \varnothing))$
axm3: $PathReduce \in (PATH \setminus \{NullPath\}) \rightarrow (BLOCK \rightarrow PATH)$
axm4: $\forall p \cdot p \in PATH \setminus \{NullPath\} \Rightarrow (\exists b \cdot b \in BLOCK \Rightarrow PathReduce(p)(b) \in PathSub[\{p\}])$
axm5: $\forall p \cdot p \in PATH \setminus \{NullPath\} \Rightarrow (\exists b \cdot b \in BLOCK \Rightarrow Path2Block[\{PathReduce(p)(b)\}] = Path2Block[\{p\}] \setminus \{b\})$
axm6: $Path2Block[\{NullPath\}] = \varnothing$
axm7: $PathIncrease \in PATH \rightarrow (BLOCK \rightarrow PATH)$
axm8: $\forall p \cdot p \in PATH \Rightarrow (\exists b \cdot b \in BLOCK \Rightarrow p \in PathSub[\{PathIncrease(p)(b)\}])$
axm9: $\forall p \cdot p \in PATH \Rightarrow (\exists b \cdot b \in BLOCK \Rightarrow Path2Block[\{p\}] \cup \{b\} = Path2Block[\{PathIncrease(p)(b)\}])$

END

CONTEXT SR2_Point
EXTENDS SR1_Block
SETS
POS
ISLOCK
CONSTANTS
POINT
Point2InitPos
Route2Point2Pos
Lock
Unlock
AXIOMS
axm1: $POINT \subseteq BLOCK$
axm2: $Route2Point2Pos \in (ROUTE \leftrightarrow POINT) \rightarrow POS$
axm4: $Point2InitPos \in POINT \rightarrow POS$
axm3: $\forall pos \cdot pos \in POS \Rightarrow (finite(POS))$
axm5: $partition(ISLOCK, \{Lock\}, \{Unlock\})$
END

CONTEXT SR3_Signal
EXTENDS SR2_Point
SETS
SIGNAL
CONSTANTS
Red
Green
AXIOMS
axm1: $partition(SIGNAL, \{Red\}, \{Green\})$
END

MACHINE SR_M0
SEES SR0_Path
VARIABLES
Route_Req
Route_Cel
Route_Occ
Route2Path
INVARIANTS
inv1: $Route\_Req \subseteq ROUTE$
inv2: $Route\_Cel \subseteq ROUTE$
inv3: $Route\_Occ \subseteq ROUTE$
inv4: $Route2Path \in ROUTE \nrightarrow PATH$
inv5: $\forall r1, r2 \cdot (r1 \neq r2 \land r1 \in dom(Route2Path) \land r2 \in dom(Route2Path)) \Rightarrow (PathConflict^{-1}[Route2Path[\{r1\}]] \cap Route2Path[\{r2\}] = \varnothing)$
inv6: $\forall r \cdot r \in Route\_Occ \Rightarrow (Route2Path[\{r\}] \neq \varnothing)$
EVENTS
Initialisation
begin
act1: $Route\_Req := \varnothing$
act2: $Route\_Cel := \varnothing$
act3: $Route\_Occ := \varnothing$
act4: $Route2Path := \varnothing$
end
Event ATS_Request ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \notin Route\_Req$
then
act1: $Route\_Req := Route\_Req \cup \{r\}$
end
Event Route_Reserve ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$
grd4: $PathConflict[Route2InitPath[\{r\}]] \cap ran(Route2Path) = \varnothing$
then
act1: $Route2Path := Route2Path \cup \{r \mapsto Route2InitPath(r)\}$
end
Event Train_Enter ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in dom(Route2Path)$
grd2: $r \notin Route\_Occ$
then
act1: $Route\_Occ := Route\_Occ \cup \{r\}$
end
Event Route_Sequential_Release ⟨ordinary⟩ $\widehat{=}$
any
r
cp
sp

where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $cp = Route2Path(r)$
grd4: $cp \neq NullPath$
grd5: $sp \in PathSub[\{cp\}]$
then
act1: $Route2Path(r) := sp$
end

Event Train_Leave ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $Route2Path(r) = NullPath$
then
act1: $Route\_Occ := Route\_Occ \setminus \{r\}$
end

Event Route_Release ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in dom(Route2Path)$
grd2: $Route2Path(r) = NullPath$
grd3: $r \notin Route\_Occ$
then
act1: $Route2Path := \{r\} \lhd Route2Path$
act2: $Route\_Req := Route\_Req \setminus \{r\}$
end

Event ATS_Cancel ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in Route\_Req$
then
act1: $Route\_Cel := Route\_Cel \cup \{r\}$
end

Event Route_Cancel ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in Route\_Req$
grd2: $r \in Route\_Cel$
grd3: $r \notin Route\_Occ$
grd4: $r \in dom(Route2Path)$
then
act1: $Route\_Req := Route\_Req \setminus \{r\}$
act2: $Route\_Cel := Route\_Cel \setminus \{r\}$
act3: $Route2Path := \{r\} \lhd Route2Path$
end

END

MACHINE SR_M1
REFINES SR_M0
SEES SR1_Block
VARIABLES
Route_Req
Route_Cel
Route_Occ
Route2Path
Block2Route
Route2OccPath
INVARIANTS
inv1: $Block2Route \in BLOCK \pfun ROUTE$
reserve
inv2: $Route2OccPath \in ROUTE \pfun PATH$
inv3: $\forall p \cdot p \in PATH \Rightarrow (\forall q \cdot q \notin PathConflict[\{p\}] \Leftrightarrow (Path2Block[\{p\}] \cap Path2Block[\{q\}] = \varnothing))$
EVENTS
Initialisation ⟨extended⟩
begin
act1: $Route\_Req := \varnothing$
act2: $Route\_Cel := \varnothing$
act3: $Route\_Occ := \varnothing$
act4: $Route2Path := \varnothing$
act5: $Block2Route := \varnothing$
act6: $Route2OccPath := \varnothing$
end
Event ATS_Request ⟨ordinary⟩ $\widehat{=}$
extends ATS_Request
any
r
where
grd1: $r \notin Route\_Req$
then
act1: $Route\_Req := Route\_Req \cup \{r\}$
end
Event Block_Reserve ⟨ordinary⟩ $\widehat{=}$
any
r
p
where
grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$
grd4: $p \in PATH$
grd5: $p = Route2InitPath(r)$
grd6: $Path2Block[\{p\}] \cap dom(Block2Route) = \varnothing$
then
act1: $Block2Route := Block2Route \cup (Path2Block[\{p\}] \times \{r\})$
end
Event Route_Reserve ⟨ordinary⟩ $\widehat{=}$
extends Route_Reserve
any
r
where
grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$

grd4: $PathConflict[Route2InitPath[\{r\}]] \cap ran(Route2Path) = \varnothing$
grd5: $Block2Route^{-1}[\{r\}] = Path2Block[\{Route2InitPath(r)\}]$

then

act1: $Route2Path := Route2Path \cup \{r \mapsto Route2InitPath(r)\}$

end

Event Train_Enter ⟨ordinary⟩ $\widehat{=}$

extends Train_Enter

any

r

where

grd1: $r \in dom(Route2Path)$
grd2: $r \notin Route\_Occ$
grd3: $r \notin dom(Route2OccPath)$

then

act1: $Route\_Occ := Route\_Occ \cup \{r\}$
act2: $Route2OccPath := Route2OccPath \cup \{r \mapsto NullPath\}$

end

Event Train_Head_Move ⟨ordinary⟩ $\widehat{=}$

any

r
op
b

where

grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd4: $b \in Path2Block[\{Route2Path(r)\}]$
grd5: $r \in dom(Route2OccPath)$
grd6: $op = Route2OccPath(r)$
grd7: $b \notin Path2Block[\{op\}]$

then

act1: $Route2OccPath(r) := PathIncrease(op)(b)$

end

Event Train_Rear_Move ⟨ordinary⟩ $\widehat{=}$

any

r
op
b

where

grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2OccPath)$
grd3: $op = Route2OccPath(r)$
grd4: $b \in Path2Block[\{op\}]$
grd5: $op \neq NullPath$

then

act1: $Route2OccPath(r) := PathReduce(op)(b)$

end

Event Block_Release ⟨ordinary⟩ $\widehat{=}$

extends Route_Sequential_Release

any

r
cp
sp
b

where

grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $cp = Route2Path(r)$
grd4: $cp \neq NullPath$

grd5: $sp \in PathSub[\{cp\}]$
grd6: $sp = PathReduce(cp)(b)$
grd7: $r \in dom(Route2OccPath)$
grd8: $b \notin Path2Block[\{Route2OccPath(r)\}]$

then

act1: $Route2Path(r) := sp$

end

Event Train_Leave ⟨ordinary⟩ $\widehat{=}$

extends Train_Leave

any

r

where

grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $Route2Path(r) = NullPath$
grd4: $Block2Route^{-1}[\{r\}] = \varnothing$

then

act1: $Route\_Occ := Route\_Occ \setminus \{r\}$

end

Event Route_Release ⟨ordinary⟩ $\widehat{=}$

extends Route_Release

any

r

where

grd1: $r \in dom(Route2Path)$
grd2: $Route2Path(r) = NullPath$
grd3: $r \notin Route\_Occ$

then

act1: $Route2Path := \{r\} \lhd Route2Path$
act2: $Route\_Req := Route\_Req \setminus \{r\}$
act3: $Block2Route := Block2Route \rhd \{r\}$
act4: $Route2OccPath := \{r\} \lhd Route2OccPath$

end

Event ATS_Cancel ⟨ordinary⟩ $\widehat{=}$

extends ATS_Cancel

any

r

where

grd1: $r \in Route\_Req$

then

act1: $Route\_Cel := Route\_Cel \cup \{r\}$

end

Event Block_Cancel ⟨ordinary⟩ $\widehat{=}$

any

r

where

grd1: $r \in Route\_Cel$
grd2: $r \in ran(Block2Route)$
grd3: $r \notin Route\_Occ$

then

act1: $Block2Route := Block2Route \rhd \{r\}$

end

Event Route_Cancel ⟨ordinary⟩ $\widehat{=}$

extends Route_Cancel

any

r

where

grd1: $r \in Route\_Req$

grd2: $r \in Route\_Cel$
grd3: $r \notin Route\_Occ$
grd4: $r \in dom(Route2Path)$
grd5: $Block2Route^{-1}[\{r\}] = \varnothing$

then

act1: $Route\_Req := Route\_Req \setminus \{r\}$
act2: $Route\_Cel := Route\_Cel \setminus \{r\}$
act3: $Route2Path := \{r\} \lhd Route2Path$

end

END

MACHINE SR_M2
REFINES SR_M1
SEES SR2_Point
VARIABLES

Route_Req
Route_Cel
Route_Occ
Route2Path
Block2Route
Route2OccPath
Point2Pos
Point2Lock

INVARIANTS

inv1: $Point2Pos \in POINT \rightarrow POS$
inv2: $Point2Lock \in POINT \rightarrow ISLOCK$
inv3: $\forall r, po \cdot po \in (Path2Block[\{Route2Path(r)\}] \cap POINT) \wedge r \in dom(Route2Path) \Rightarrow (Point2Pos(po) = Route2Point2Pos(\{r \mapsto po\}))$
inv4: $\forall r, po \cdot po \in (Path2Block[\{Route2Path(r)\}] \cap POINT) \wedge r \in dom(Route2Path) \Rightarrow (Point2Lock(po) = Lock)$

EVENTS

Initialisation ⟨extended⟩
begin
act1: $Route\_Req := \varnothing$
act2: $Route\_Cel := \varnothing$
act3: $Route\_Occ := \varnothing$
act4: $Route2Path := \varnothing$
act5: $Block2Route := \varnothing$
act6: $Route2OccPath := \varnothing$
act7: $Point2Pos := Point2InitPos$
act8: $Point2Lock := POINT \times \{Unlock\}$
end

Event ATS_Request ⟨ordinary⟩ $\widehat{=}$
extends ATS_Request
any
r
where
grd1: $r \notin Route\_Req$
then
act1: $Route\_Req := Route\_Req \cup \{r\}$
end

Event Block_Reserve ⟨ordinary⟩ $\widehat{=}$
extends Block_Reserve
any
r
p
where
grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$
grd4: $p \in PATH$
grd5: $p = Route2InitPath(r)$
grd6: $Path2Block[\{p\}] \cap dom(Block2Route) = \varnothing$
then
act1: $Block2Route := Block2Route \cup (Path2Block[\{p\}] \times \{r\})$
end

Event Point_Switch ⟨ordinary⟩ $\widehat{=}$
Before Route_Reserve
any
po
r
where
grd1: $r \notin Route\_Cel$
grd2: $r \notin dom(Route2Path)$
grd3: $po \in Block2Route^{-1}[\{r\}] \cap POINT$
grd4: $Point2Pos(po) \neq Route2Point2Pos(\{r \mapsto po\})$
grd5: $Point2Lock(po) = Unlock$
then
act1: $Point2Pos(po) := Route2Point2Pos(\{r \mapsto po\})$
end

Event Point_Lock ⟨ordinary⟩ $\widehat{=}$
Before Route_Reserve
any
r
po
where
grd1: $r \notin Route\_Cel$
grd2: $r \notin dom(Route2Path)$
grd3: $po \in Block2Route^{-1}[\{r\}] \cap POINT$
grd4: $Point2Pos(po) = Route2Point2Pos(\{r \mapsto po\})$
grd5: $Point2Lock(po) = Unlock$
then
act1: $Point2Lock := \{po \mapsto Lock\} \lhd Point2Lock$
end

Event Route_Reserve ⟨ordinary⟩ $\widehat{=}$
extends Route_Reserve
any
r
where
grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$
grd4: $PathConflict[Route2InitPath[\{r\}]] \cap ran(Route2Path) = \varnothing$
grd5: $Block2Route^{-1}[\{r\}] = Path2Block[\{Route2InitPath(r)\}]$
grd6: $\forall po \cdot (po \in (Path2Block[Route2InitPath[\{r\}]] \cap POINT)) \Rightarrow (Point2Pos(po) = Route2Point2Pos(\{r \mapsto po\}))$
grd7: $\forall po \cdot (po \in (Path2Block[Route2InitPath[\{r\}]] \cap POINT)) \Rightarrow (Point2Lock(po) = Lock)$
then
act1: $Route2Path := Route2Path \cup \{r \mapsto Route2InitPath(r)\}$
end

Event Train_Enter ⟨ordinary⟩ $\widehat{=}$
extends Train_Enter
any
r
where
grd1: $r \in dom(Route2Path)$
grd2: $r \notin Route\_Occ$
grd3: $r \notin dom(Route2OccPath)$
then
act1: $Route\_Occ := Route\_Occ \cup \{r\}$
act2: $Route2OccPath := Route2OccPath \cup \{r \mapsto NullPath\}$
end

Event Train_Head_Move ⟨ordinary⟩ $\widehat{=}$
extends Train_Head_Move

any
r
op
b
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd4: $b \in Path2Block[\{Route2Path(r)\}]$
grd5: $r \in dom(Route2OccPath)$
grd6: $op = Route2OccPath(r)$
grd7: $b \notin Path2Block[\{op\}]$
then
act1: $Route2OccPath(r) := PathIncrease(op)(b)$
end

Event Train_Rear_Move ⟨ordinary⟩ $\widehat{=}$
extends Train_Rear_Move
any
r
op
b
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2OccPath)$
grd3: $op = Route2OccPath(r)$
grd4: $b \in Path2Block[\{op\}]$
grd5: $op \neq NullPath$
then
act1: $Route2OccPath(r) := PathReduce(op)(b)$
end

Event Point_Unlock_Release ⟨ordinary⟩ $\widehat{=}$
Before Block_Release
any
r
cp
sp
po
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $cp = Route2Path(r)$
grd4: $sp = PathReduce(cp)(po)$
grd5: $r \in dom(Route2OccPath)$
grd7: $po \in POINT$
grd8: $po \notin Path2Block[\{Route2Path(r)\}]$
then
act1: $Point2Lock(po) := Unlock$
end

Event Block_Release ⟨ordinary⟩ $\widehat{=}$
extends Block_Release
any
r
cp
sp
b
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $cp = Route2Path(r)$
grd4: $cp \neq NullPath$

grd5: $sp \in PathSub[\{cp\}]$
grd6: $sp = PathReduce(cp)(b)$
grd7: $r \in dom(Route2OccPath)$
grd8: $b \notin Path2Block[\{Route2OccPath(r)\}]$

then

act1: $Route2Path(r) := sp$

end

Event Train_Leave ⟨ordinary⟩ $\widehat{=}$

extends Train_Leave

any

r

where

grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $Route2Path(r) = NullPath$
grd4: $Block2Route^{-1}[\{r\}] = \varnothing$

then

act1: $Route\_Occ := Route\_Occ \setminus \{r\}$

end

Event Route_Release ⟨ordinary⟩ $\widehat{=}$

extends Route_Release

any

r

where

grd1: $r \in dom(Route2Path)$
grd2: $Route2Path(r) = NullPath$
grd3: $r \notin Route\_Occ$

then

act1: $Route2Path := \{r\} \lhd Route2Path$
act2: $Route\_Req := Route\_Req \setminus \{r\}$
act3: $Block2Route := Block2Route \rhd \{r\}$
act4: $Route2OccPath := \{r\} \lhd Route2OccPath$

end

Event ATS_Cancel ⟨ordinary⟩ $\widehat{=}$

extends ATS_Cancel

any

r

where

grd1: $r \in Route\_Req$

then

act1: $Route\_Cel := Route\_Cel \cup \{r\}$

end

Event Point_Unlock_Cancel ⟨ordinary⟩ $\widehat{=}$

Before Train_Enter

any

r
po

where

grd1: $r \in Route\_Cel$
grd2: $r \notin Route\_Occ$
grd3: $po \in Block2Route^{-1}[\{r\}] \cap POINT$
grd4: $Point2Lock(po) = Lock$

then

act1: $Point2Lock := \{po \mapsto Unlock\} \lhd\!\!+ Point2Lock$

end

Event Block_Cancel ⟨ordinary⟩ $\widehat{=}$

extends Block_Cancel

any

r

where

grd1: $r \in Route\_Cel$

grd2: $r \in ran(Block2Route)$

grd3: $r \notin Route\_Occ$

grd4: $\forall po \cdot po \in Block2Route^{-1}[\{r\}] \cap POINT \Rightarrow Point2Lock(po) = Unlock$

then

act1: $Block2Route := Block2Route \rhd \{r\}$

end

Event Route_Cancel ⟨ordinary⟩ $\widehat{=}$

extends Route_Cancel

any

r

where

grd1: $r \in Route\_Req$

grd2: $r \in Route\_Cel$

grd3: $r \notin Route\_Occ$

grd4: $r \in dom(Route2Path)$

grd5: $Block2Route^{-1}[\{r\}] = \varnothing$

then

act1: $Route\_Req := Route\_Req \setminus \{r\}$

act2: $Route\_Cel := Route\_Cel \setminus \{r\}$

act3: $Route2Path := \{r\} \lhd Route2Path$

end

END

MACHINE SR_M3
REFINES SR_M2
SEES SR3_Signal
VARIABLES
Route_Req
Route_Cel
Route_Occ
Route2Path
Block2Route
Route2OccPath
Point2Pos
Point2Lock
Route2Signal
INVARIANTS
inv1: $Route2Signal \in ROUTE \to SIGNAL$
inv2: $\forall r \cdot r \in dom(Route2OccPath) \land Route2OccPath(r) \neq NullPath \Rightarrow Route2Signal(r) = Red$
EVENTS
Initialisation
begin
act1: $Route\_Req := \varnothing$
act2: $Route\_Cel := \varnothing$
act3: $Route\_Occ := \varnothing$
act4: $Route2Path := \varnothing$
act5: $Block2Route := \varnothing$
act6: $Route2OccPath := \varnothing$
act7: $Point2Pos := Point2InitPos$
act8: $Point2Lock := POINT \times \{Unlock\}$
act9: $Route2Signal := ROUTE \times \{Red\}$
end
Event ATS_Request ⟨ordinary⟩ $\widehat{=}$
extends ATS_Request
any
r
where
grd1: $r \notin Route\_Req$
then
act1: $Route\_Req := Route\_Req \cup \{r\}$
end
Event Block_Reserve ⟨ordinary⟩ $\widehat{=}$
extends Block_Reserve
any
r
p
where
grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$
grd4: $p \in PATH$
grd5: $p = Route2InitPath(r)$
grd6: $Path2Block[\{p\}] \cap dom(Block2Route) = \varnothing$
then
act1: $Block2Route := Block2Route \cup (Path2Block[\{p\}] \times \{r\})$
end
Event Point_Switch ⟨ordinary⟩ $\widehat{=}$
extends Point_Switch
any

po
r

where

grd1: $r \notin Route\_Cel$
grd2: $r \notin dom(Route2Path)$
grd3: $po \in Block2Route^{-1}[\{r\}] \cap POINT$
grd4: $Point2Pos(po) \neq Route2Point2Pos(\{r \mapsto po\})$
grd5: $Point2Lock(po) = Unlock$

then

act1: $Point2Pos(po) := Route2Point2Pos(\{r \mapsto po\})$

end

Event Point_Lock ⟨ordinary⟩ $\widehat{=}$
extends Point_Lock

any

r
po

where

grd1: $r \notin Route\_Cel$
grd2: $r \notin dom(Route2Path)$
grd3: $po \in Block2Route^{-1}[\{r\}] \cap POINT$
grd4: $Point2Pos(po) = Route2Point2Pos(\{r \mapsto po\})$
grd5: $Point2Lock(po) = Unlock$

then

act1: $Point2Lock := \{po \mapsto Lock\} \lhd Point2Lock$

end

Event Route_Reserve ⟨ordinary⟩ $\widehat{=}$
extends Route_Reserve

any

r

where

grd1: $r \in Route\_Req$
grd2: $r \notin Route\_Cel$
grd3: $r \notin dom(Route2Path)$
grd4: $PathConflict[Route2InitPath[\{r\}]] \cap ran(Route2Path) = \varnothing$
grd5: $Block2Route^{-1}[\{r\}] = Path2Block[\{Route2InitPath(r)\}]$
grd6: $\forall po \cdot (po \in (Path2Block[Route2InitPath[\{r\}]] \cap POINT)) \Rightarrow (Point2Pos(po) = Route2Point2Pos(\{r \mapsto po\}))$
grd7: $\forall po \cdot (po \in (Path2Block[Route2InitPath[\{r\}]] \cap POINT)) \Rightarrow (Point2Lock(po) = Lock)$

then

act1: $Route2Path := Route2Path \cup \{r \mapsto Route2InitPath(r)\}$

end

Event Signal_Green_Reserve ⟨ordinary⟩ $\widehat{=}$

any

r

where

grd1: $r \in dom(Route2Path)$
grd2: $r \in Route\_Req$
grd3: $r \notin Route\_Cel$
grd4: $Block2Route^{-1}[\{r\}] = Path2Block[\{Route2InitPath(r)\}]$
grd5: $\forall po \cdot (po \in (Path2Block[Route2InitPath[\{r\}]] \cap POINT)) \Rightarrow (Point2Pos(po) = Route2Point2Pos(\{r \mapsto po\}))$
grd6: $\forall po \cdot (po \in (Path2Block[Route2InitPath[\{r\}]] \cap POINT)) \Rightarrow (Point2Lock(po) = Lock)$
grd7: $r \notin Route\_Occ$
grd8: $r \notin dom(Route2OccPath)$

then

act1: $Route2Signal := \{r \mapsto Green\} \lhd Route2Signal$

end

Event Train_Enter ⟨ordinary⟩ $\widehat{=}$

extends Train_Enter
any
r
where
grd1: $r \in dom(Route2Path)$
grd2: $r \notin Route\_Occ$
grd3: $r \notin dom(Route2OccPath)$
grd4: $Route2Signal(r) = Green$
then
act1: $Route\_Occ := Route\_Occ \cup \{r\}$
act2: $Route2OccPath := Route2OccPath \cup \{r \mapsto NullPath\}$
end
Event Signal_Red_Occupied ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in dom(Route2Path)$
grd2: $r \notin Route\_Occ$
grd3: $Route2Signal(r) = Green$
grd4: $r \in dom(Route2OccPath)$
grd5: $Route2OccPath(r) = NullPath$
then
act1: $Route2Signal(r) := Red$
end
Event Train_Head_Move ⟨ordinary⟩ $\widehat{=}$
extends Train_Head_Move
any
r
op
b
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd4: $b \in Path2Block[\{Route2Path(r)\}]$
grd5: $r \in dom(Route2OccPath)$
grd6: $op = Route2OccPath(r)$
grd7: $b \notin Path2Block[\{op\}]$
grd8: $Route2Signal(r) = Red$
then
act1: $Route2OccPath(r) := PathIncrease(op)(b)$
end
Event Train_Rear_Move ⟨ordinary⟩ $\widehat{=}$
extends Train_Rear_Move
any
r
op
b
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2OccPath)$
grd3: $op = Route2OccPath(r)$
grd4: $b \in Path2Block[\{op\}]$
grd5: $op \neq NullPath$
then
act1: $Route2OccPath(r) := PathReduce(op)(b)$
end
Event Point_Unlock_Release ⟨ordinary⟩ $\widehat{=}$
extends Point_Unlock_Release

any
r
cp
sp
po
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $cp = Route2Path(r)$
grd4: $sp = PathReduce(cp)(po)$
grd5: $r \in dom(Route2OccPath)$
grd7: $po \in POINT$
grd8: $po \notin Path2Block[\{Route2Path(r)\}]$
grd9: $Route2Signal(r) = Red$
then
act1: $Point2Lock(po) := Unlock$
end

Event Block_Release ⟨ordinary⟩ $\widehat{=}$
extends Block_Release
any
r
cp
sp
b
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $cp = Route2Path(r)$
grd4: $cp \neq NullPath$
grd5: $sp \in PathSub[\{cp\}]$
grd6: $sp = PathReduce(cp)(b)$
grd7: $r \in dom(Route2OccPath)$
grd8: $b \notin Path2Block[\{Route2OccPath(r)\}]$
then
act1: $Route2Path(r) := sp$
end

Event Train_Leave ⟨ordinary⟩ $\widehat{=}$
extends Train_Leave
any
r
where
grd1: $r \in Route\_Occ$
grd2: $r \in dom(Route2Path)$
grd3: $Route2Path(r) = NullPath$
grd4: $Block2Route^{-1}[\{r\}] = \varnothing$
then
act1: $Route\_Occ := Route\_Occ \setminus \{r\}$
end

Event Route_Release ⟨ordinary⟩ $\widehat{=}$
extends Route_Release
any
r
where
grd1: $r \in dom(Route2Path)$
grd2: $Route2Path(r) = NullPath$
grd3: $r \notin Route\_Occ$
then
act1: $Route2Path := \{r\} \lhd Route2Path$

act2: $Route\_Req := Route\_Req \setminus \{r\}$
act3: $Block2Route := Block2Route \rhd \{r\}$
act4: $Route2OccPath := \{r\} \lhd Route2OccPath$

end

Event ATS_Cancel ⟨ordinary⟩ $\widehat{=}$
extends ATS_Cancel

any
r
where
grd1: $r \in Route\_Req$
then
act1: $Route\_Cel := Route\_Cel \cup \{r\}$
end

Event Signal_Red_Cancel ⟨ordinary⟩ $\widehat{=}$
any
r
where
grd1: $r \in Route\_Cel$
grd2: $Route2Signal(r) = Green$
grd3: $r \in dom(Route2Path)$
grd4: $r \notin Route\_Occ$
then
act1: $Route2Signal := \{r \mapsto Red\} \lhd Route2Signal$
end

Event Point_Unlock_Cancel ⟨ordinary⟩ $\widehat{=}$
extends Point_Unlock_Cancel
any
r
po
where
grd1: $r \in Route\_Cel$
grd2: $r \notin Route\_Occ$
grd3: $po \in Block2Route^{-1}[\{r\}] \cap POINT$
grd4: $Point2Lock(po) = Lock$
grd5: $Route2Signal(r) = Red$
then
act1: $Point2Lock := \{po \mapsto Unlock\} \lhd Point2Lock$
end

Event Block_Cancel ⟨ordinary⟩ $\widehat{=}$
extends Block_Cancel
any
r
where
grd1: $r \in Route\_Cel$
grd2: $r \in ran(Block2Route)$
grd3: $r \notin Route\_Occ$
grd4: $\forall po \cdot po \in Block2Route^{-1}[\{r\}] \cap POINT \Rightarrow Point2Lock(po) = Unlock$
then
act1: $Block2Route := Block2Route \rhd \{r\}$
end

Event Route_Cancel ⟨ordinary⟩ $\widehat{=}$
extends Route_Cancel
any
r
where
grd1: $r \in Route\_Req$
grd2: $r \in Route\_Cel$
grd3: $r \notin Route\_Occ$

grd4: $r \in dom(Route2Path)$
grd5: $Block2Route^{-1}[\{r\}] = \varnothing$

then

act1: $Route\_Req := Route\_Req \setminus \{r\}$
act2: $Route\_Cel := Route\_Cel \setminus \{r\}$
act3: $Route2Path := \{r\} \lhd Route2Path$

end

END